CLAIMS

What is claimed is:

1. A method of converting a logically ordered character stream into a character stream suitable for display by a computer and comprehension by a user, said logically ordered character stream having a plurality of characters and control codes contained within it, said method comprising:

assigning in a functional programming language bidirectional attributes to the logical character stream;

assigning in a functional programming language initial level numbers and honoring any directional overrides by explicit processing;

changing attribute types based upon surrounding attribute types through weak and neutral processing in a functional programming language;

associating final level numbers to the logical character stream through implicit processing in a functional programming language; and

reordering said characters within said logical character stream according to said final level numbers such that said reordered characters form a character stream in display order, said reordering being performed in a functional programming language.

2. The method as set forth in Claim 1 wherein said step of assigning bidirectional attributes further comprises

obtaining said bidirectional attributes from a character database.

3. The method as set forth in Claim 1 further comprising the step of grouping characters into sequential runs using type constructors and level such that characters are processed collectively rather than individually.

4. The method as set forth in Claim 1 wherein said step of changing attribute types based upon surrounding attribute types through weak and neutral processing in a functional programming language comprises providing blocks of functional programming language indexed by name weak type processing, neutral type processing, and implicit level processing such that said method may be readily used as a reference.

5. The method as set forth in Claim 1 wherein one or more steps are provided in Haskell functional language.

6. The method as set forth in Claim 1 wherein one or more steps are provided in Erlang functional language.

7. The method as set forth in Claim 1 wherein one or more steps are provided in SML functional language.

8. The method as set forth in Claim 1 wherein one or more steps are provided in Miranda functional language.

9. The method as set forth in Claim 1 wherein one or more steps are provided in Lisp functional language.

10. The method as set forth in Claim 1 wherein one or

more steps are provided in Scheme functional language.

11. A computer readable medium encoded with software causing a computer to perform the following actions:

receiving a logically ordered character stream;

assigning in a functional programming language bidirectional attributes to the logical character stream;

assigning in a functional programming language initial level numbers and honoring any directional overrides by explicit processing;

changing attribute types based upon surrounding attribute types through weak and neutral processing in a functional programming language;

associating final level numbers to the logical character stream through implicit processing in a functional programming language; and

reordering said characters within said logical character stream according to said final level numbers such that said reordered characters form a character stream in display order, said reordering being performed in a functional programming language.

12. The computer readable medium as set forth in Claim 11 wherein said software for performing said assignment of bidirectional attributes further comprises software for obtaining said bidirectional attributes from a character database.

13. The computer readable medium as set forth in Claim 11 wherein said software further comprises software for performing grouping characters into sequential runs using type constructors and level such that characters are processed collectively rather than individually.

14. The computer readable medium as set forth in Claim 11 wherein said software for performing the action of changing attribute types based upon surrounding attribute types through weak and neutral processing in a functional programming language comprises software organized into blocks of functional programming language indexed by name weak type processing, neutral type processing, and implicit level processing such that said method may be readily used as a reference.

15. The computer readable medium as set forth in Claim 11 wherein said software is Haskell functional language.

16. The computer readable medium as set forth in Claim 11 wherein said software is Erlang functional language.

17. The computer readable medium as set forth in Claim 11 wherein said software is SML functional language.

18. The computer readable medium as set forth in Claim 11 wherein said software is Miranda functional language.

19. The computer readable medium as set forth in Claim 11 wherein said software is Lisp functional language.

20. The computer readable medium as set forth in Claim 11 wherein said software is Scheme functional language.

21. A text code conversion system for converting logically ordered text streams and displaying said text streams in a display order, said system comprising:

a character stream receiver for receiving a logically ordered character stream;

a bidirectional attribute assignor realized in a functional programming language for assigning bidirectional attributes to a received logical character stream;

an initial level assignor realized in a functional programming language for assigning initial level numbers and for honoring any directional overrides by explicit processing;

an attribute type changer realized in a functional programming language for changing attribute types based upon surrounding attribute types through weak and neutral;

a final level assignor realized in a functional programming language for associating final level numbers to the logical character stream through implicit processing; and

a character resequencer realized in a functional programming language for reordering said characters within said logical character stream according to said final level numbers such that said reordered characters form a character stream in display order.

22. The text code conversion system as set forth in Claim 21 wherein said bidirectional attributes assignor is adapted to obtain said bidirectional attributes from a character database.

23. The text code conversion system as set forth in Claim 21 further comprising a character grouper for grouping characters into sequential runs using type constructors and level such that characters are processed collectively rather than individually.

24. The text code conversion system as set forth in Claim 21 wherein said attribute type changer attribute type changer comprises blocks of functional programming language indexed by name weak type processing, neutral type processing, and implicit level processing such that said method may be readily used as a reference.

25. The text code conversion system as set forth in Claim 21 wherein said bidirectional attribute assignor, initial level assignor, attribute type changer, final level assignor, and character resequencer comprise Haskell functional language.

26. The text code conversion system as set forth in Claim 21 wherein said bidirectional attribute assignor, initial level assignor, attribute type changer, final level assignor, and character resequencer comprise Erlang functional language.

27. The text code conversion system as set forth in

Claim 21 wherein said bidirectional attribute assignor, initial level assignor, attribute type changer, final level assignor, and character resequencer comprise SML functional language.

28. The text code conversion system as set forth in Claim 21 wherein said bidirectional attribute assignor, initial level assignor, attribute type changer, final level assignor, and character resequencer comprise Miranda functional language.

29. The text code conversion system as set forth in Claim 21 wherein said bidirectional attribute assignor, initial level assignor, attribute type changer, final level assignor, and character resequencer comprise Lisp functional language.

30. The text code conversion system as set forth in Claim 21 wherein said bidirectional attribute assignor, initial level assignor, attribute type changer, final level assignor, and character resequencer comprise Scheme functional language.